

AutoShield

Simulink 기반 자동차 부품 연계보안 기술 연구

부산대학교 컴퓨터공학과

 팀원: 홍재왕, 레풍푸, 석재영

 지도교수: 손준영

목차

1 연구 배경 및 목표

2 기존 문제점

3 연구 방법론

4 시스템 아키텍처

5 송·수신부 구조

6 보안 알고리즘 구현





7 실험 및 결과

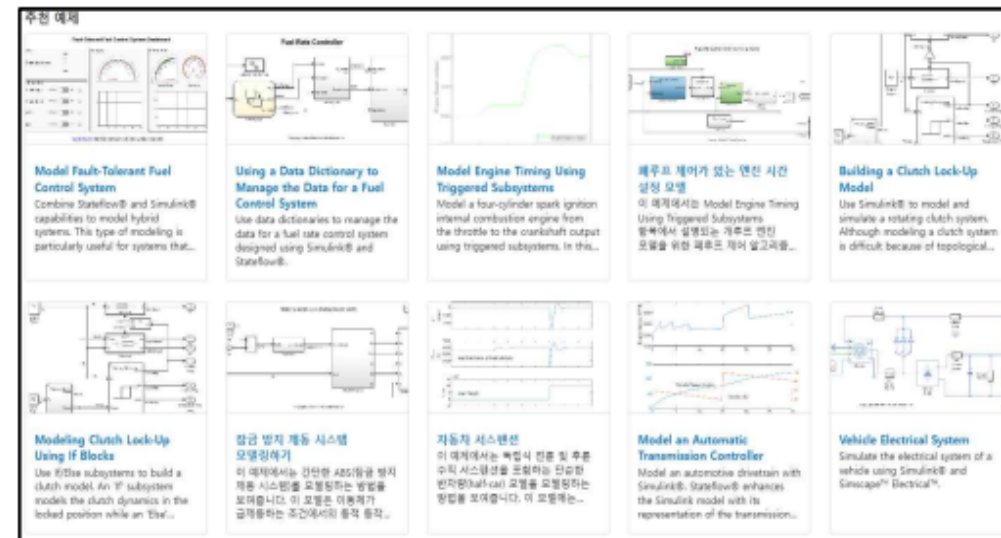
8 결론 및 전망

연구 배경

- 🛡️ **현대 자동차 네트워크의 보안 위협 증가** - 컴퓨터화 및 네트워크 확장으로 차량 내부 통신의 해킹 위험성 급증
- ⚡ **CAN 및 Ethernet 취약점** - 2015년 Jeep Cherokee 해킹 사례처럼 기존 차량 프로토콜의 인증/암호화 부재로 인한 보안 위험
- 🔧 **실환경 실험의 한계** - 실제 차량 환경에서의 보안 검증은 비용/위험 부담이 큼
- 🖥️ **Simulink 기반 가상 실험의 필요성** - 다양한 보안 알고리즘의 설계·적용·비교가 가능한 안전한 테스트 환경 구축 필요

연구 목표

-  **Simulink 환경에서 ECU 통신 상황 완전 재현** - MATLAB/Simulink 플랫폼을 통해 자동차 전장부품(ECU) 간 통신 모델링 구현
-  **하이브리드 보안 체계적 적용** - CAN-FD 및 Ethernet을 활용한 통합 보안 프레임워크 구축, HMAC-SHA256과 Freshness Counter 기반 이중 검증
-  **실시간 UI 통합 모니터링** - 시각화 대시보드를 통한 데이터 흐름 및 보안 상태 실시간 모니터링, 공격 시나리오 테스트 기능
-  **미래 확장성 기반 구축** - 다양한 보안 알고리즘 비교·검증, AI/경량암호 확장 가능한 프레임워크 설계

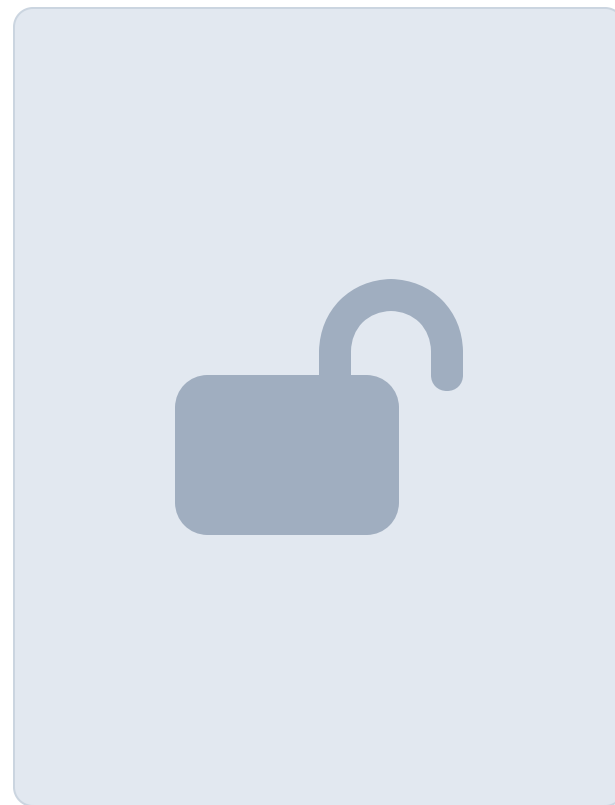


Simulink Library



기존 문제점

- ⚠ **CAN 프로토콜의 취약점** - 인증/암호화 메커니즘 결여, 브로드캐스트 방식으로 인한 스니핑/스푸핑 공격에 취약
- 🔌 **Ethernet의 한계** - 외부 연결성으로 인한 공격 표면 확대, 통합 네트워크 환경에서의 보안 경계 설정 난제
- 🔧 **실차 실험의 위험성** - 물리적 위험, 고비용, 장비 제약으로 인한 다양한 시나리오 테스트 어려움
- ⚙️ **보안 알고리즘 적용의 복잡성** - ECU 성능/대역폭 제약, 다양한 통신 프로토콜 간 호환성 문제

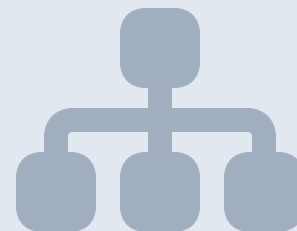


차량 통신 네트워크의 취약점과 보안 위협 모델



연구 방법론

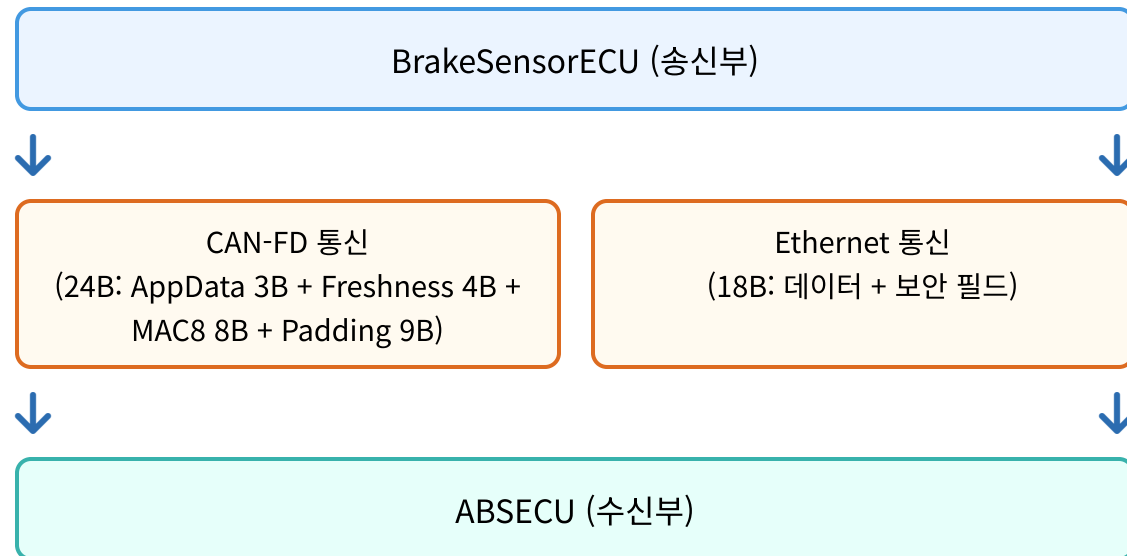
- ⚙️ **Simulink와 MATLAB 환경** - 차량 네트워크 시뮬레이션과 보안 알고리즘 검증을 위한 통합 개발 환경 활용
- 🚗 **Vehicle Network Toolbox** - CAN-FD 및 Ethernet 프로토콜 통신을 위한 전문 라이브러리 활용, 통신 시나리오 설계
- 🔗 **워크플로우 설계** - 센서 데이터 생성 → 암호화 → 전송 → 복호화 → 출력 시각화 전체 흐름의 체계적 구현
- ✓ **단위테스트 및 검증** - 암호화/복호화, 신호 흐름 구조화, 결과 분석 및 비교 등 단계별 테스트 적용



Simulink 기반 자동차 부품 연계보안 기술 워크플로우

시스템 아키텍처 개요

- 🔧 **하이브리드 통신 구조** - CAN-FD와 Ethernet을 병렬로 활용한 송신부(TX)와 수신부(RX) 구성
- 🛡️ **이중 경로 보안** - 두 프로토콜에 모두 HMAC-SHA256과 Freshness Counter 적용으로 보안성 강화
- 🔧 **ECU 간 통신 모델링** - BrakeSensorECU(송신)와 ABSECU(수신) 간 안전한 데이터 전송 체계 구축
- 🔒 **Fail-Safe 구조** - 보안 검증 실패 시 데이터를 ECU 로직에 전달하지 않는 안전 메커니즘 구현

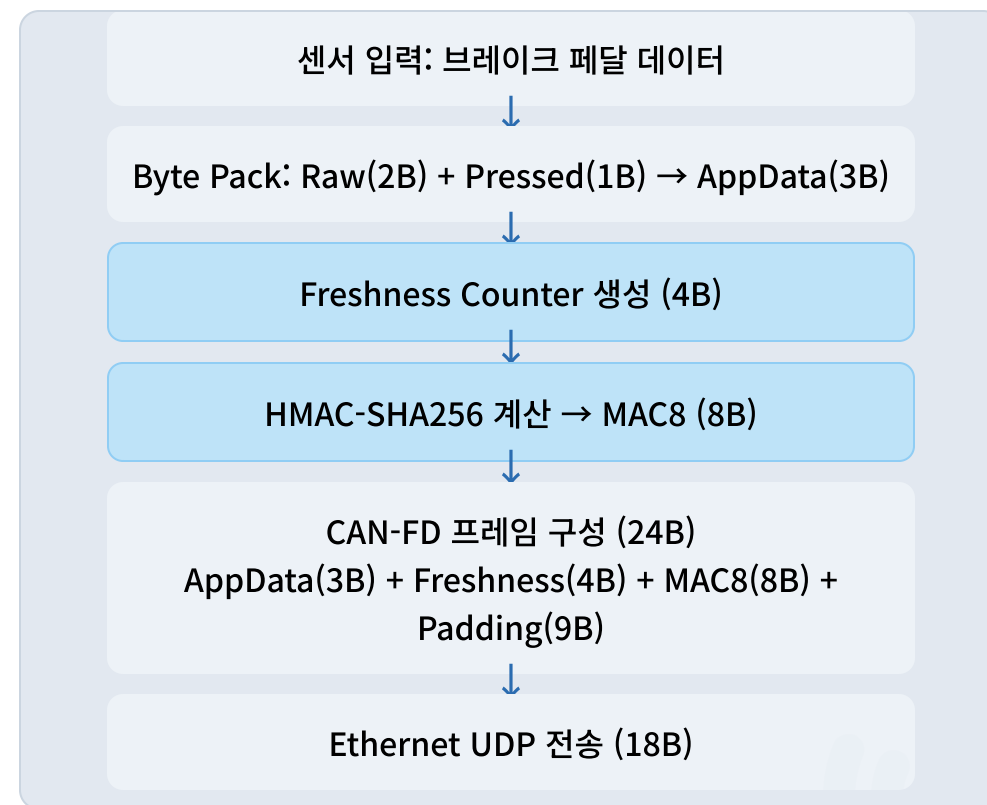


AutoShield 시스템 하이브리드 통신 아키텍처 구조도 (FIG10, FIG11, FIG13 참조)



송신부(BrakeSensorECU) 구조

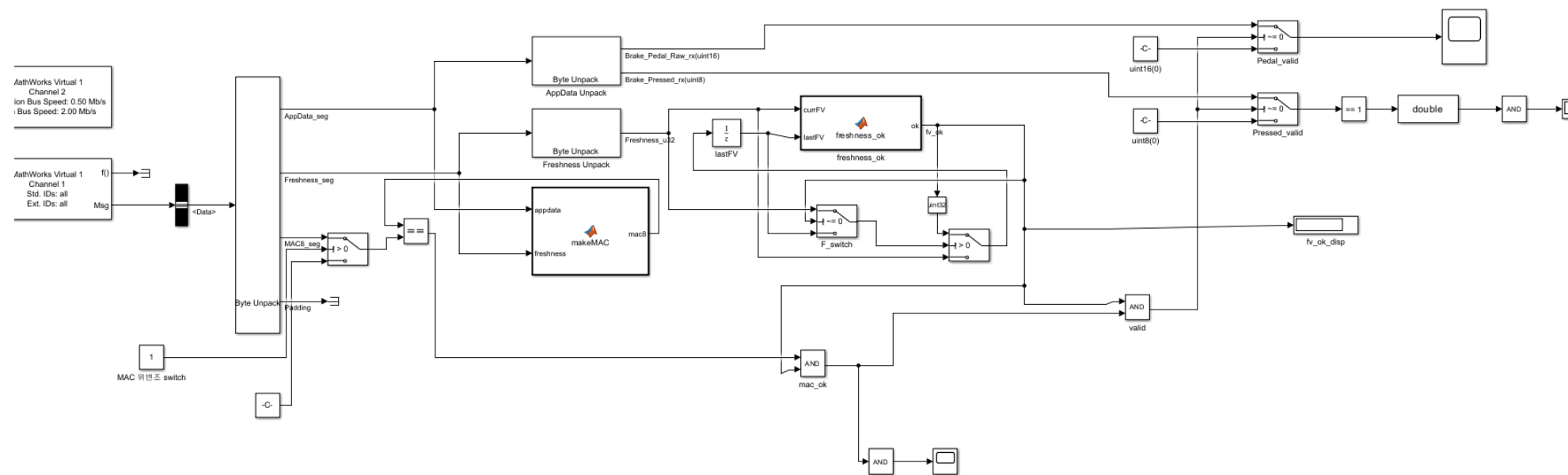
- 🔊 **센서 시나리오 생성** - 슬라이더(Brake Pedal Raw) 및 스위치(Pressed)로 브레이크 페달 입력 상태 시뮬레이션
- 📦 **데이터 패킹 과정** - Raw 데이터(2B) + Pressed 상태(1B)를 결합하여 3B AppData 생성
- 🛡️ **보안 데이터 생성** - Freshness Counter(4B) 생성 및 HMAC-SHA256 연산(32B 중 상위 8B만 사용)
- 📶 **이중 전송 경로** - CAN-FD(24B 프레임) 및 Ethernet(18B UDP) 통신으로 병렬 전송



송신부(BrakeSensorECU)의 데이터 처리 및 전송 흐름

수신부(ABSECU) 구조

- 프레임 해체 및 필드 분리 - CAN-FD 24바이트 프레임에서 AppData(3B), Freshness(4B), MAC8(8B) 필드를 분리하여 검증 처리
- 이중 검증 시스템 - HMAC-SHA256 기반 MAC 검증 및 Freshness Counter 검증을 순차적으로 수행, 둘 다 통과(valid)해야만 ECU 로직에 연결
- Fail-Safe 구조 구현 - 데이터 위변조 또는 재전송 공격 탐지 시 철저히 ECU 로직에서 차단하여 잠재적 위험 방지
- Variant Source/Switch 활용 - CAN-FD/Ethernet 경로를 자유롭게 전환하며 같은 검증 로직 적용



수신부 데이터 검증 플로우

🛡️ **이중 검증 체계** - 모든 수신 데이터는 두 단계의 보안 검증을 통과해야만 ECU 로직에 전달됨

🔒 **1차 검증: HMAC-SHA256** - 데이터 무결성 검증

- AppData + Freshness에 대한 HMAC 값 재계산
- 수신된 MAC8 값과 비교하여 일치 여부 확인

🔄 **2차 검증: Freshness Counter** - 재전송 공격 방지

- 현재 수신된 Freshness 값(currFV) \geq 마지막 값(lastFV)
- 차이 값 \leq 윈도우(W=1024) 조건 확인

✔️ **Fail-Safe 구현** - 두 검증 모두 통과(valid=1)일 때만 ECU 로직에 연결, 그 외에는 완전 차단

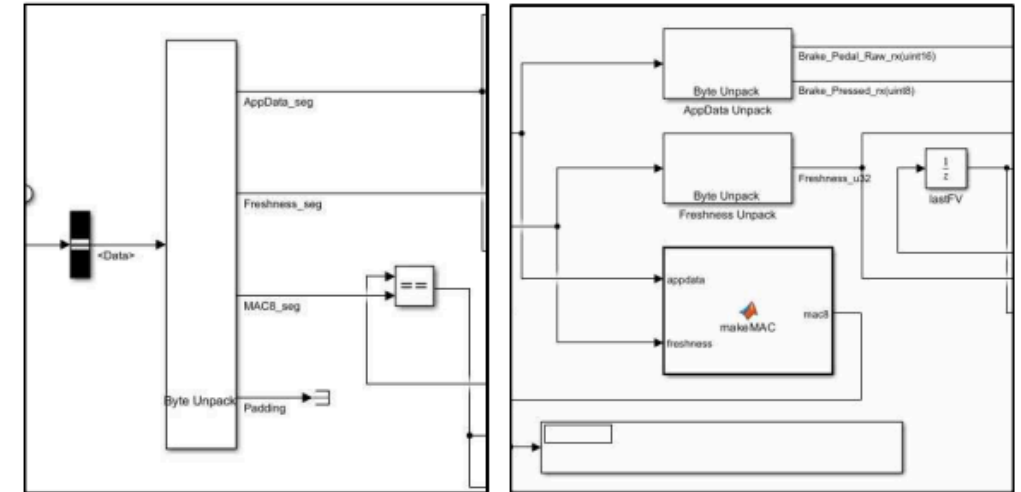


FIG14, 15. 수신부(RX) 1

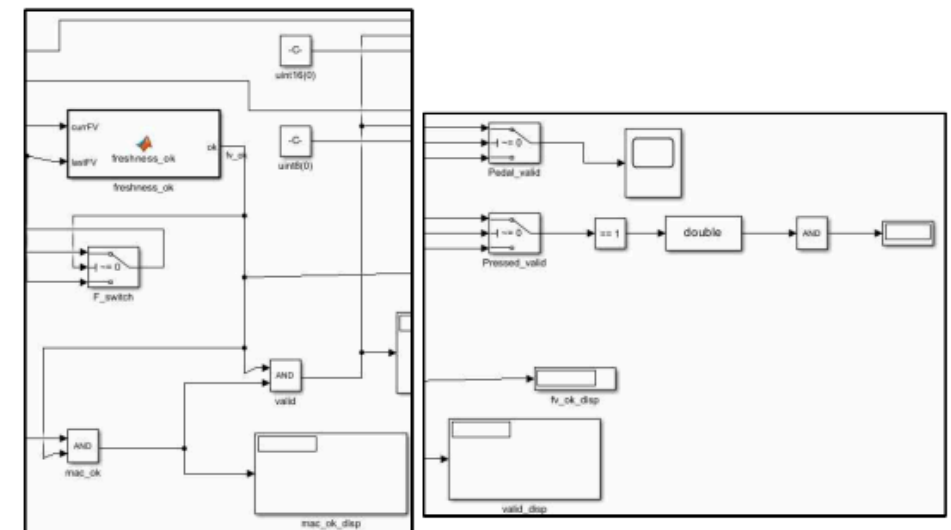


FIG16, 17. 수신부(RX) 2



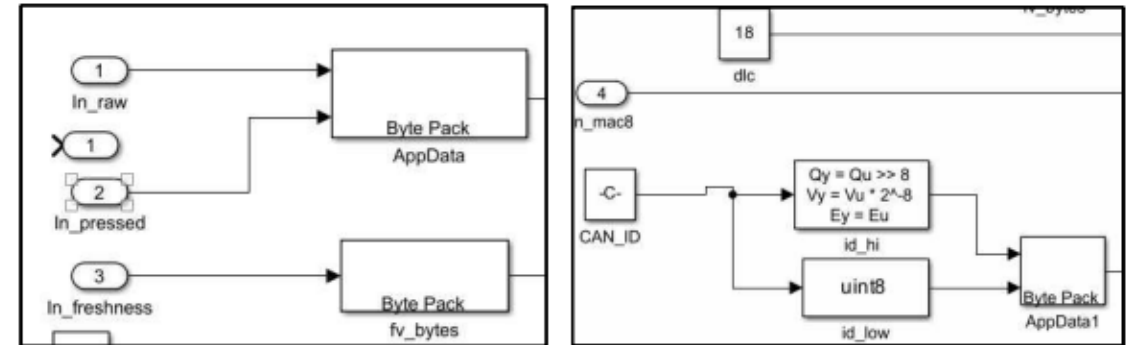
송신부/이더넷(ETH_TX) 구조

품 UDP/ETH 기반 18바이트 통신 프레임 - AppData 3B + Freshness 4B + MAC8 8B + 부가정보 3B로 구성된 효율적인 통신 구조

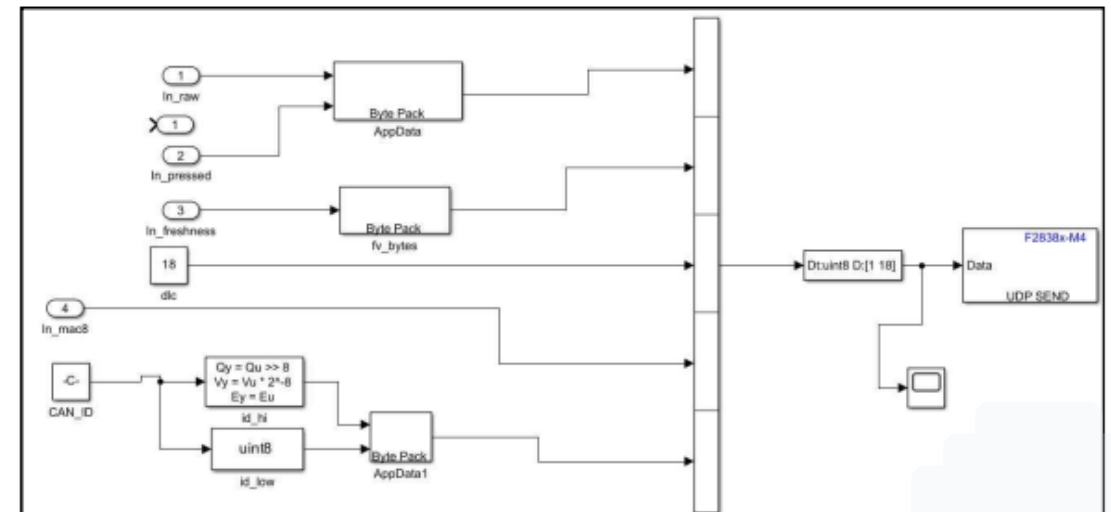
↔ Outport 신호 처리 방식 - 송신부에서 생성된 4개의 신호 (AppData, Freshness, MAC8, Valid)를 병렬로 처리하여 ETH PDU 구성

</> Concatenate 블록 활용 - ID/DLC 등을 포함한 18B PDU 생성 후 UDP Send 블록을 통한 전송

↻ CAN-FD와 병렬 운용 - 동일 데이터를 다중 경로로 전송하여 보안성과 가용성 향상



ETH_TX 블록 다이어그램 구조



이더넷 통신 흐름도 및 구현 결과

HMAC-SHA256 구현

- 🔒 **입력 데이터 처리** - appdata(3B) + freshness(4B) 총 7바이트 메시지에 대한 HMAC-SHA256 연산 수행
- ✂ **트렁케이션(Truncation) 기법** - 32바이트 HMAC-SHA256 해시값 중 상위 8바이트만 MAC 값으로 사용하여 대역폭 효율성 확보
- ⌘ **Simulink 코드 생성 친화적 설계** - 고정 크기 버퍼 사용, 루프 경계 명확화, 메모리 관리 최적화
- 🔑 **키(Key) 관리** - 안전한 키 저장 및 키 갱신 매커니즘 구현으로 장기적 보안성 확보



Freshness Counter 함수 구현

- ↻ **재전송 공격(Replay Attack) 방지** - 이전에 캡처된 정상 메시지의 재사용을 방지하기 위한 신선도 검증 메커니즘
- ↑ **단순하고 효율적인 검증 로직** - $\text{currFV} \geq \text{lastFV}$ 조건 및 차이값 \leq 윈도우($W=1024$) 제한으로 연산 효율성 보장
- 🛡 **실시간성과 보안성의 균형** - 최소한의 연산으로 리플레이 공격 탐지 및 정상 시 오차 허용
- ⚠ **Fail-Safe 구조** - Freshness 검증 실패 시 해당 ECU 로직 차단으로 안전성 확보

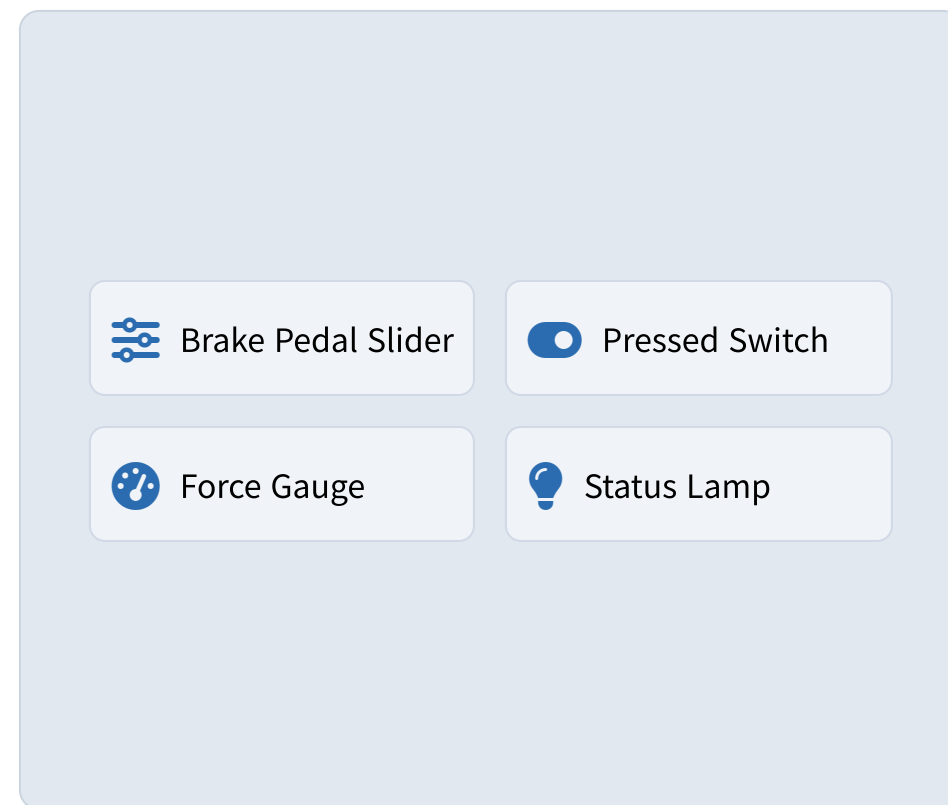
```
% Freshness Counter 검증 함수
function [valid, newLastFV] = validateFreshness(currFV,
lastFV, W)
% 초기 검증 상태: 유효하지 않음
valid = false;
newLastFV = lastFV;

% 현재 FV가 마지막 FV보다 크거나 같은지 확인
if currFV >= lastFV
% 허용 윈도우(W) 내에 있는지 확인
if currFV - lastFV <= W
valid = true;
newLastFV = currFV;
end
end
end
```

Freshness Counter 검증 함수 구현 (FIG26)

실시간 UI & 대시보드 설계

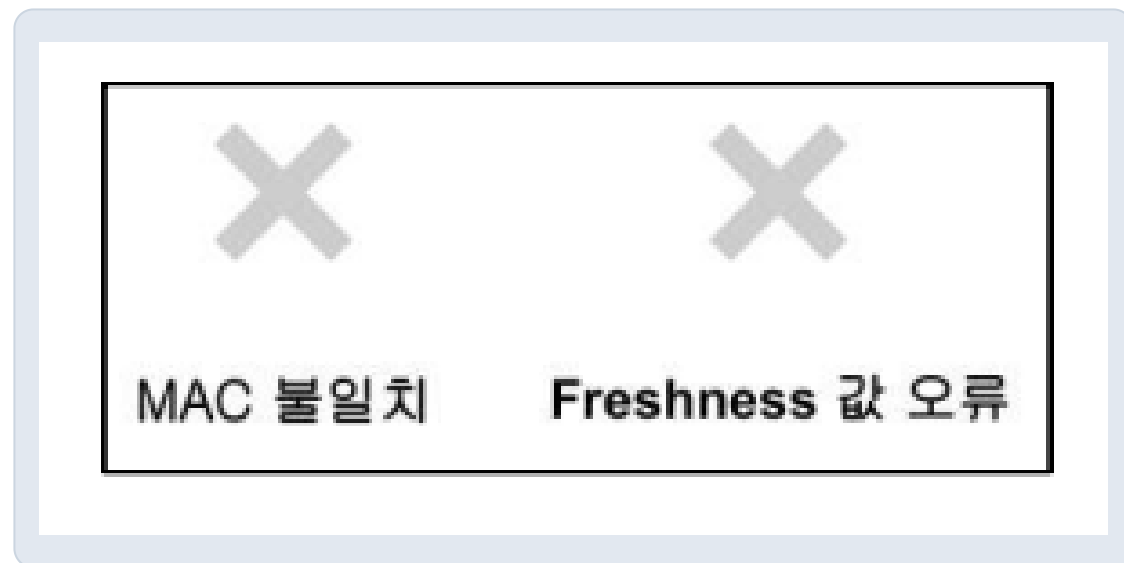
- 🖥️ **Simulink Dashboard 구성** - 직관적인 슬라이더(Brake Pedal Raw)와 스위치(Pressed) 컴포넌트로 다양한 브레이크 입력 시나리오 시뮬레이션
- 🕒 **실시간 상태 모니터링** - 수신부 상태 시각화를 위한 게이지(Brake Force) 및 램프(Brake Status)로 즉각적 피드백 제공
- ⚠️ **공격 시나리오 통합** - MAC 위변조 및 Freshness 오류 시뮬레이션 버튼으로 다양한 보안 공격 상황 재현 가능
- ✅ **검증 결과 시각화** - 정상(체크마크), MAC 오류(빨간색 X), Freshness 오류(회색 X) 등 3가지 상태로 명확한 결과 표시



Simulink Dashboard 및 App Designer를 활용한 실시간 모니터링 UI

공격 시나리오 테스트

- ✓ **정상 시나리오** - 모든 검증(MAC, Freshness) 통과, 브레이크 센서 데이터가 ABSECU에 정상 반영, UI에서 램프 ON 상태로 표시
- ✗ **MAC 오류 시나리오** - 송신 데이터 위변조 시, 수신부에서 MAC 불일치로 인한 검증 실패, 램프 OFF 상태 유지
- 🔄 **Freshness 오류 시나리오** - 재전송 공격 또는 윈도우 초과 시, Freshness Counter 검증 실패, 램프 OFF 상태 유지
- ⚙️ **Pressed 변수 검증** - Pressed=0으로 설정 시 ECU 로직에서 비활성화, 다양한 입력값에 대한 시스템 반응 테스트

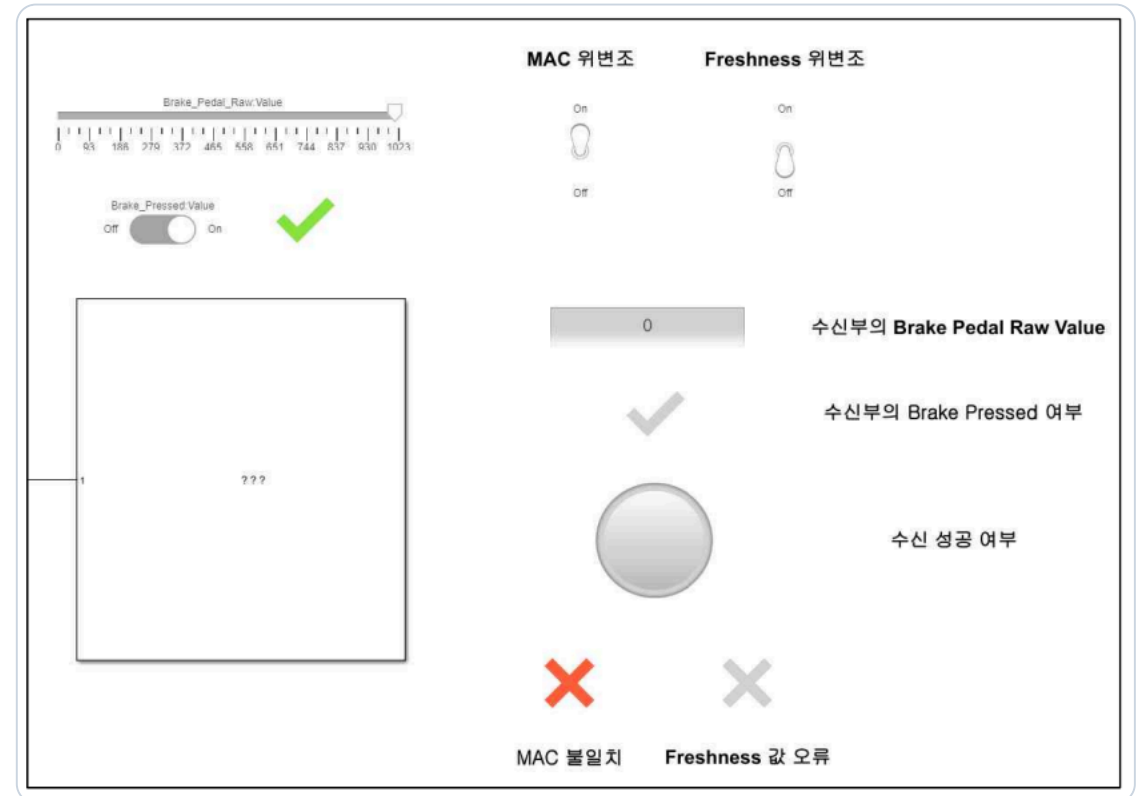


실험 결과: 보안성 검증

- ❖ **하이브리드 통신 환경 검증** - CAN-FD와 Ethernet 경로 모두에서 보안 알고리즘이 정상 작동하는 것을 확인
- 🔍 **데이터 위·변조 탐지** - HMAC-SHA256 검증을 통해 모든 MAC 불일치 패킷을 100% 차단
- 🔄 **재전송 공격 방지** - Freshness Counter 검증으로 윈도우를 벗어난 모든 재전송 패킷 탐지
- 🚫 **Fail-Safe 로직 실효성** - 검증 실패 시 해당 데이터가 ECU에 전달되지 않는 안전 메커니즘 확인

핵심 검증 결과

- ✓ CAN-FD 24B 프레임 내 AppData, Freshness, MAC8 효과적 연계 검증
- ✓ 송·수신 간 실시간성 유지하며 보안 알고리즘 정상 작동
- ✓ 다양한 공격 시나리오에서 Fail-Safe 동작 100% 확인



결론 및 향후 연구

✓ **연구 성과:** CAN-FD 24바이트 프레임 내 AppData, Freshness, MAC8을 포함한 연계검증 체계를 성공적으로 구축하여 데이터 위·변조 및 재전송 공격을 효과적으로 차단

↻ **하이브리드 통신:** CAN-FD와 Ethernet 병렬 환경에서 동일한 보안성을 유지하며, Fail-Safe 시스템과 실시간 시각화 UI 구현 완료

향후 연구 방향:

🔒 TLS/DTLS 적용

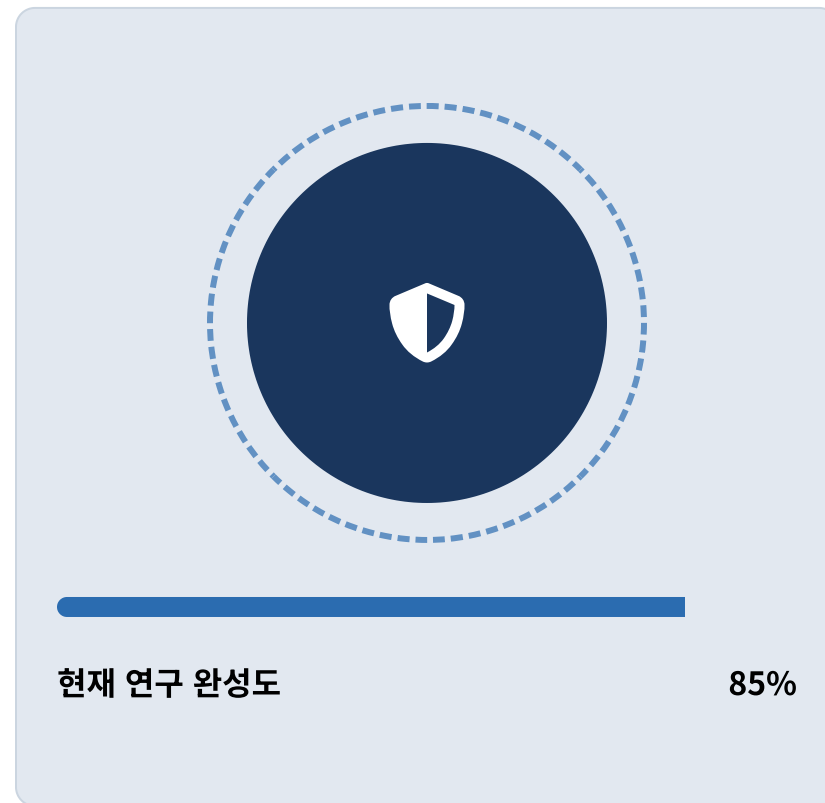
⚙️ 다중 ECU 확장

🧠 AI 기반 이상탐지

🚗 실차 HIL 평가

🛡️ 경량암호 비교

🎓 교육/실무 활용



AutoShield 프로젝트: 하이브리드 보안 통신 체계 확장 가능성

감사합니다

